# STATISTICAL CHARACTERIZATION OF PROTEIN ENSEMBLES

By

**Diego Rother**
**Guillermo Sapiro**
and
**Vijay Pande**

**IMA Preprint Series # 2103**

( March 2006 )

INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS
UNIVERSITY OF MINNESOTA
400 Lind Hall
207 Church Street S.E.
Minneapolis, Minnesota 55455–0436

Phone: 612/624-6066     Fax: 612/626-7370
URL: http://www.ima.umn.edu

| | |
|---|---|
| **Report Documentation Page** | *Form Approved*<br>*OMB No. 0704-0188* |

| | | |
|---|---|---|
| 1. REPORT DATE<br>**MAR 2006** | 2. REPORT TYPE | 3. DATES COVERED<br>**00-00-2006 to 00-00-2006** |

| | |
|---|---|
| 4. TITLE AND SUBTITLE<br>**Statistical Characterization of Protein Ensembles (PREPRINT)** | 5a. CONTRACT NUMBER |
| | 5b. GRANT NUMBER |
| | 5c. PROGRAM ELEMENT NUMBER |
| 6. AUTHOR(S) | 5d. PROJECT NUMBER |
| | 5e. TASK NUMBER |
| | 5f. WORK UNIT NUMBER |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>**University of Minnesota,Institute for Mathematics and its Applications,207 Church Street SE,Minneapolis,MN,55455-0436** | 8. PERFORMING ORGANIZATION REPORT NUMBER |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) | 10. SPONSOR/MONITOR’S ACRONYM(S) |
| | 11. SPONSOR/MONITOR’S REPORT NUMBER(S) |

12. DISTRIBUTION/AVAILABILITY STATEMENT

**Approved for public release; distribution unlimited**

13. SUPPLEMENTARY NOTES

14. ABSTRACT

**When accounting for structural fluctuations or measurement errors, a single rigid structure may not be sufficient to represent a protein. One approach to solve this problem is to represent the possible conformations as a discrete set of observed conformations, an ensemble. In this work, we follow a different richer approach, and introduce a framework for estimating probability density functions in very high dimensions, and then apply it to represent ensembles of folded proteins. This proposed approach combines techniques such as kernel density estimation, maximum likelihood, cross-validation, and bootstrapping. We present the underlying theoretical and computational framework and apply it to artificial data and protein ensembles obtained from molecular dynamics simulations, and compare the results with those obtained experimentally, illustrating the potential and advantages of this representation.**

15. SUBJECT TERMS

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT<br>**unclassified** | b. ABSTRACT<br>**unclassified** | c. THIS PAGE<br>**unclassified** | **Same as Report (SAR)** | **31** | |

# Statistical Characterization of Protein Ensembles

Diego Rother, [1,2] Guillermo Sapiro,[1] and Vijay Pande[3]

## Abstract

When accounting for structural fluctuations or measurement errors, a single rigid structure may not be sufficient to represent a protein. One approach to solve this problem is to represent the possible conformations as a discrete set of observed conformations, an ensemble. In this work, we follow a different richer approach, and introduce a framework for estimating probability density functions in very high dimensions, and then apply it to represent ensembles of folded proteins. This proposed approach combines techniques such as kernel density estimation, maximum likelihood, cross-validation, and bootstrapping. We present the underlying theoretical and computational framework and apply it to artificial data and protein ensembles obtained from molecular dynamics simulations, and compare the results with those obtained experimentally, illustrating the potential and advantages of this representation.

# 1 Introduction and motivation

A single structure is often used to represent a protein. This can be considered natural when the structure is assumed to be rigid, but protein structure is known to fluctuate under physiological conditions. This fact, overlooked in many situations in favor of the simplicity of a single structure, may be advantageously accounted for at times when high resolution techniques for structure determination are available. Even if the "true" structure *were* fixed and unique, the uncertainty in its determination by the (imperfect) measurement of some property (i.e., diffraction, magnetic resonance, etc.), also produces variability, since those methods generally optimize a model to fit the observations, a process prone to find multiple local minima. A similar situation arises when simulations are used for structure "determination," the modeled energy landscape is populated with multiple local minima. As a result of this and other intrinsic simulation characteristics (e.g., randomness), multiple structure representatives for the same protein are possible.

In applications where the fluctuations can not be ignored, and moreover, are to be favorably exploited, how should they be represented and incorporated into the calculations? One way is to represent the protein structure not as a single conformation but as a finite set of conformations, corresponding to different "observations" of its state. In this work we propose a different richer approach, consisting of estimating a probability density function (pdf) from the available observations of the state, and using this pdf to represent the ensemble (a finite set of conformations is just a particular case of this, with the pdf being delta functions placed in the observation points). This rich representation is starting to gain interest in the protein research community, e.g., [1, 2] and Lindorff-Larsen (personal communication). For example, this type of representation has been recently pursued to rank the space of conformations in agreement with NMR observations [1].

[1] Department of Electrical and Computer Engineering, University of Minnesota, 200 Union St. SE, Minneapolis, MN 55455, USA. diroth,guille@ece.umn.edu

[2] To whom correspondence should be addressed.

[3] Chemistry Department, Stanford University, Stanford, CA 94305-5080, USA. pande@stanford.edu

This representation may allow one to see aspects previously hidden when the ensemble was regarded as a set of discrete conformations (e.g., the modes), and also provides a natural framework to perform certain operations, e.g., compare ensembles of the same protein that were obtained by different methods, or determine the probability that a particular conformation belongs to the ensemble [3]. New conformations that combine properties of the ensemble can be obtained as well from the pdf.

A possible further application of the framework comes from the observation that multiple local minima lie close to the global minimum, as postulated to explain the robustness of the native state, and suggested as a way to predict it for certain protein classes [4]. This observation, if general, can be translated into the requirement that the native structure (or ensemble) must reside where the density of local minima is high.

Further motivation for the necessity to understand the conformational space and its probability distribution is suggested by recent work toward high-resolution de novo structure prediction [5], see in particular the authors remark that "conformational sampling remains the primary stumbling block" toward this challenging goal. It is imperative then to have a good description of the ensemble, and ideally, such description is given by a probability density function.

Important by-products of the approach here introduced include an idea of the completeness of the sample to represent the space of conformations that the protein adopts, an estimate of the conformational entropy (and its error) which may have important thermodynamic consequences (Lindorff-Larsen, personal communication); and a measurement of the dependency between variables.

It is thereby clearly supported by the current efforts in protein research the need for a good understanding of the protein conformational space, and in particular, of its probability density function (pdf). It is the primary goal of this paper to present a theoretical and computational framework to compute such a probability density function.

Protein ensembles consist of conformations usually having hundreds or even thousands degrees of freedom. How can inferences be made from samples sizes that are roughly of the same order? This challenging question is addressed in this article. We derive, under clear optimality criteria from information theory, the best possible pdf from the available data. To achieve this, we decompose the global density as a product of lower dimensional factors, conditional probabilities themselves, chosen by a genetic algorithm to maximize the global likelihood. The approach exploits the fact that each degree of freedom (or coordinate) does not strongly depend on every other coordinate, but only on a few, which are automatically found by our approach. Then, we proceed to estimate each factor using classical density estimation techniques, that is, Kernel Density Estimation and Maximum Likelihood. In addition to computing the probability distribution of the ensemble, we explicitly and automatically obtain the critical dependencies between the variables, such as torsion angles.

The main data used in this work comes from simulations of protein ensembles obtained by means of molecular dynamics [6]. The framework here described can be used to characterize other protein ensembles, computed either via molecular dynamics or using other structural determination method (e.g., rotameric libraries with applications in high-resolution protein folding [7], or even direct multiple physical measurements). The framework can also be used to include protein flexibility in protein docking [8]. More on this will be presented in the discussion section.

The remainder of this paper is organized as follows. In Section 2 we give a description of the mathematical and computational method proposed to compute the desired probability density function, given a finite set of conformations. As a proof of concept and for pedagogic reasons, we first use the developed framework on an artificial dataset. This is presented in Section 3.1. In Section 3.2 we use the framework in real data. In addition to computing the pdf, we explicitly derive the critical inner dependencies of the torsion angles, and produce novel conformations sampled from the computed pdf. Their relationship with experimental data is studied as well. Concluding remarks and discussions are provided in Section 4.

## 2 Methods

An ensemble[4] is a set of conformations of the same protein. Each conformation corresponds to a particular arrangement of the protein's constitutive atoms in three dimensional (3D) space. This arrangement can be described (or partially described) by different sets of features depending on the application at hand. In this work, we consider the backbone of the protein, which can be completely described by the usual 2($M$-1) torsion angles (($M$-1) $\phi$'s and ($M$-1) $\psi$'s) where $M$ is the number of residues in the protein [9].

Our goal is to develop a technique to estimate the density of the unknown process that generates the set of conformations, the ensemble. This density is to be estimated from its available samples (a finite set of conformations represented by vectors of length 2($M$-1)). To address this we consider that a coordinate of the sample conformation is related to just a few other coordinates, without knowing in advance to which ones. In other words, we set out to infer the relationships between the coordinates (torsion angles in our example), and use this information to estimate the density of the process more efficiently. While this is a natural assumption based on the chemical nature of proteins, it is also fundamental to reduce the dimensionality of the problem, which is needed due to the existence of only finite and relatively few observations.

The proposed computational framework involves a number of components in the fields of statistics, information theory, artificial neural networks, and computer science. It is philosophically related to Hinton's Products of Experts (PoE) [10], in the sense that several low dimensional probability densities ("experts" in Hinton's terminology), each one able only to explain local features of the dataset, are composed (multiplied) to explain global features. These approaches are best at explaining global features from local details, but can not in general handle the effects of global features on local details. Our approach differs from Hinton's in that the experts are independent by (automatic) construction, thus avoiding the need to renormalize the product. An additional difference between the approaches is that Hinton uses parametric models for the experts, while we estimate its shape directly from the data. Further restrictions on the experts in our approach also guarantee the expert's heterogeneity, and assure that all the local features are considered in the construction of the global density.

Our approach is also related to Akaike's Information Criterion (AIC) [11, 12], in that the choice of the order of the model selected is based in the Kullback-Leibler distance to the "true" unknown probability density. Contrary to AIC, in our approach the number of parameters does not explicitly appears in the criterion, but only through the degree of smoothing applied. This has

[4] To avoid the confusion derived from using the word "sample" to denote a "set of conformations" and also a "single conformation from that set" we reserve this word for the first meaning ("a set of conformations"). We also use "ensemble" for the same concept. We use the words "observation" and "point" for the second meaning ("a single conformation").

the advantage that each parameter is not equally weighted, but it is weighted according to the particular role it plays in the model.

In this section we present the proposed density estimation framework, and the rationale behind the selection of the particular methods to fulfill each task. For that purpose, and for easy reference and completeness, a brief review of each relevant method is included. In Section 2.1, a procedure for estimating a density is introduced. In Section 2.2, we analyze the errors and limits of this estimation procedure. Finally, in Section 2.3, we extend the technique for the kind of dataset of interest (conformation ensembles). To avoid obscuring the main concepts, the non critical implementation details are omitted from this article. They can be obtained, together with the code, from the authors by request.

## 2.1 Density estimation

### 2.1.1 Maximum likelihood principle

In the search for the "best" density estimate, the best way to start is to define what is "best," or at least, what is "better." The field of statistics provides a tool for that purpose: the *maximum likelihood principle* [13].

The rationale behind the maximum likelihood principle can be stated quite simple: from all the "possible" models (or densities) that could have generated the ensemble, select the most probable one. More formally, let $S=\{\vec{x}_1,\vec{x}_2,...,\vec{x}_N\}$ be an ensemble of $N$ independent and identically distributed observations in $\Re^{2(M-1)}$, generated by one of the models in $\mathcal{M}$, the set of all "possible" models (formally defined in 2.1.2). Without going into further details at this point, let us mention that both the data points $\vec{x}_i$ and the models belong to a continuous space. This is the reason to use densities instead of probabilities throughout these explanations. Let $M_h \in \mathcal{M}$ be a model in the family parameterized by *h*, a parameter to be detailed in Section 2.1.2. Then, using Bayes' law, the probability density that the model generated the ensemble can be expressed as:

$$p(M_h \mid S) = p(S \mid M_h).\frac{p(M_h)}{p(S)}$$

where:

$p(S \mid M_h)$ is the probability density that the model $M_h$ generates the ensemble $S$. We will refer to it as the *ensemble (or sample) likelihood* and we will denote it by $L_h(S)$. Since the observations in $S$ are assumed to be independent, this term can be easily computed (now the model is assumed to be known), $L_h(S)\overset{\Delta}{=}P(S \mid M_h)=\prod_{i=1}^{N}P(\vec{x}_i \mid M_h)=\prod_{i=1}^{N}f_h(\vec{x}_i)$, where $f_h$ is the density corresponding to the model $M_h$;

$p(S)$ is the unconditioned probability density of the ensemble. It is of little interest here, since it is equal for all the models and therefore as is commonly done, we will ignore it; and

$p(M_h)$ is the *a priori* probability density of the model, knowledge that should bias the choice of one model over another. When such information is not available, or if it is not reliable, the common practice is to assume that all models are equally probable. For the datasets dealt with in this article, *a priori* information dictated by the physics of the process *is* available (see [14]). Nevertheless, we choose for simplicity not to include this information in the model at this stage, keeping in mind that the results can be improved by doing otherwise. Note that this *a priori* probability is defined over the space of models $\mathcal{M}$.

Consequently, the factor $\frac{p(M_h)}{p(S)}$ is the same for all models (when all models are equally possible), and the most probable model is simply the one that makes the data most probable, maximizes the sample likelihood. Simplification can be obtained by maximizing the logarithm of this quantity instead,

$$\log(L_h(S)) = \log P(S \mid M_h) = \sum_{i=1}^{N} \log f_h(\vec{x}_i)$$

Since the logarithm function is monotonically increasing, it attains the maximum at the same model but will have a much simpler derivative.

It will be useful for later developments to note the close relationship between the log-likelihood and the *empirical entropy* or *finite sample average* of entropy,[5] defined as [15]:

$$H_S(f) \overset{\Delta}{=} -\frac{1}{N}\sum_{i=1}^{N} \log(f(\vec{x}_i)) \tag{1}$$

Its relation to the log-likelihood is readily apparent:

$$H_S(f) = -\frac{1}{N}\log(L_h(S))$$

Then, maximizing log-likelihood is equivalent to minimizing the empirical entropy, and since we find the notation simpler and the concepts richer, we choose to work with the latter.

An identical result can be obtained through a completely different (at first sight) approach using the *relative entropy*, also known in the literature as the Kullback-Leibler divergence, cross entropy, or asymmetric divergence [15]. It is defined, for two densities *f*(*x*) and *g*(*x*), as

$$D(f\|g) = \int f(\vec{x}).\log\left(\frac{f(\vec{x})}{g(\vec{x})}\right) d\vec{x}$$

The relative entropy is a measure of the "distance" between two distributions.[6] Consequently it seems natural to define the best density estimate ( $\hat{f}(\vec{x})$ ) as the one that minimizes the distance to the true unknown density ( $f(\vec{x})$ ). The new score to minimize is [13]:

$$D(f\|\hat{f}) = \int f(\vec{x})\log\left(\frac{f(\vec{x})}{\hat{f}(\vec{x})}\right) d\vec{x} = \int f(\vec{x})\log(f(\vec{x}))\, d\vec{x} - \int f(\vec{x})\log(\hat{f}(\vec{x}))\, d\vec{x} =$$
$$= -H(f) - E_f\left(\log(\hat{f}(\vec{x}))\right) \approx -H(f) - E_S\left(\log(\hat{f}(\vec{x}))\right) = -H(f) + H_S(\hat{f})$$

Since the first term in the last expression is constant, the expression to minimize is identical to the one that already was found in Equation (1). The maximum likelihood model is the one that is closer to the true density, as measured by the relative entropy "distance." The "approximately

[5] Since the empirical entropy converges to the entropy $\left(\overset{\Delta}{=}\int f(\vec{x})\log(f(\vec{x}))\,d\vec{x}\right)$ as the sample size grows [34], we may abuse language and use the terms as synonyms.

[6] Strictly speaking it is not a distance since it is not symmetric and does not satisfy the triangle inequality. Nonetheless, it is always non-negative and zero if and only if *f* = *g*, and for this reason it is often useful to think of it as a "distance" between distributions [15].

equal" symbol in the last derivation entails that, when a finite sample is used, the score obtained is only an approximation to the true score for the model. The implications of this fact are discussed in Section 2.2.

Different metrics could have been chosen to measure the discrepancy between the estimated and true densities, and each choice would have resulted in a different (optimal) estimate. Our choice, based on classical information theory and statistics, tries to capture the order of the true density (reflected by a function of the *quotient* of the two densities being integrated) rather than its absolute value (as would be the case for the $L_p$ norm, where a function of the *difference* between the densities is integrated). An additional advantage of this choice is that it leads to more tractable calculations.

As Viola points out [13], there are three main reasons why maximum likelihood may fail to find an accurate model: a) There are no sufficiently accurate models in the considered set of possible models $\mathcal{M}$. For this reason it is important to impose only the weakest assumptions on the density, in our case smoothness, Section 2.1.2; b) The search for the best model may fail to discover the model that globally minimizes the entropy (because it gets stuck in a local minima), even though it belongs to the considered set of possible models $\mathcal{M}$, hence the importance of a good optimization algorithm; and c) Unlikely observations drawn from a model are only improbable, not impossible. If an unlikely sample is drawn from a model, it could well be assigned to another model that makes it more likely. This risk becomes smaller as the sample size grows.

### 2.1.2 The hypothesis space

Having provided a way to compare models, and from this to select the best one, the next step is to define the set of "possible" models $\mathcal{M}$, or *hypothesis space*, from where the best model should be chosen. Since we do not want to loose generality at this point by restricting our attention to a particular kind of densities, we adopt the sample-based approach, where the sample itself defines the model. In particular, we are interested in the method of *Kernel Density Estimation*, also known as *Parzen Window Density Estimation* [16]. According to this method, in one dimension the density estimates are given by

$$\hat{f}_h(x,S) = \frac{1}{Nh}.\sum_{i=1}^{N} K\left(\frac{x-x_i}{h}\right) \tag{2}$$

where $K(x)$[7] is a probability density function known as the *kernel function* and $h$, apart from indexing the space, is the *window width*, also known as the *bandwidth* or the *smoothing parameter*. An alternative but otherwise equivalent expression can be given in terms of the convolution of the sample with the kernel, $\hat{f}_h(x,S) = \frac{1}{h}K\left(\frac{x}{h}\right) * \left(\frac{1}{N}\sum_{i=1}^{N}\delta(x-x_i)\right)$.

The role of the kernel is to "spread" the mass of the observations around its original position. Usually, $K$ is a unimodal even function, falling off quickly to zero. Bell shaped functions in general, and Gaussians in particular, are frequently used kernels. In this work we use the Von Mises kernel, which plays the role of the Gaussian density for angular data [17].

[7] To simplify the exposition we assumed that the kernel is "stretched" by the bandwidth. For periodic kernels (in $[0,2\pi)$ in our case) clearly this approach does not work, and the shape of the kernel must change as well when the bandwidth changes. See [17] for a detailed discussion.

What was just presented in one dimension is easily generalized to higher dimensions. In *d*-dimensions, the equation for the kernel approximation (analogous to Equation (2)) is

$$\hat{f}_h(\vec{x},S)=\frac{1}{Nh^d}.\sum_{i=1}^{N}K\left(\frac{\vec{x}-\vec{x}_i}{h}\right) \tag{3}$$

### 2.1.3 Crossvalidation

It was stated in Section 2.1.1 that one of the reasons that might lead to the maximum likelihood criterion to perform poorly is the absence of adequate models in the hypothesis space. This could tempt the naïve user to include as many models as possible in that set. In particular, for the definition of the set of functions that we gave in the previous section, this means that no constrains are imposed on the bandwidth. This is not recommendable, let's see why.

When the same sample is used both to approximate the density function *and* to estimate the entropy (likelihood), the expression for the entropy (from equations (1) and (3)) becomes:

$$H_S\left(\hat{f}_h\right)=-\frac{1}{N}\sum_{i=1}^{N}\log\left(\hat{f}_h(\vec{x}_i,S)\right)=-\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{Nh^d}.\sum_{j=1}^{N}K\left(\frac{\vec{x}_i-\vec{x}_j}{h}\right)\right) \tag{4}$$

Remember that Equation (4) assigns a score to each model/density $\hat{f}_h$. The lower the score, the better the model. The problem of density estimation can then be stated as finding the model, or in other words finding the bandwidth *h*, that minimizes Equation (4). Unfortunately, this expression has no minimum for $h\in(0,\infty)$, and as it tends to minus infinity, the corresponding selected model gets further and further away from the desired model. Since the mass of the kernel sits around the origin and falls off quickly as we move away from it, $\sum_{j=1}^{N}K\left(\frac{\vec{x}_i-\vec{x}_j}{h}\right)\xrightarrow[h\to 0]{}K\left(\frac{\vec{x}_i-\vec{x}_i}{h}\right)=K(\vec{0})$, and $H_S\left(\hat{f}_h\right)\xrightarrow[h\to 0]{}-\log\left(\frac{K(\vec{0})}{Nh^d}\right)\xrightarrow[h\to 0]{}-\infty$. As a result, the density estimate tends to the "function" that has one delta in each one of the sample points. The solution was "over-trained" to fit the data, disregarding any previous knowledge about the true density we may have had. This stresses the importance of carefully choosing the hypothesis space. What is known about the functions should be included in the definition of the hypothesis space to avoid overfitting, but not "overdoing it," risking to exclude the correct density from the hypothesis space.

A possible escape from this situation is to set a lower limit for the bandwidth *h*. But, how to select this limit in a sensible way is far from trivial. Furthermore, this limit should depend on the sample size and the unknown density itself. We rather use another approach instead, known as *cross-validation* or *leave-one-out* [13]. Alternative approaches for estimating the bandwidth can be found in [16, 18].

Since the problem originated for using the same sample twice (both to construct the density function *and* to estimate the entropy by evaluating the density in the sample points), the cross-validation technique splits the sample in two (or uses two different samples if possible), and uses one part to construct the density and the other to estimate the entropy.

It may seem at first as a waste of data to use some sample points to compute the entropy, instead of using them to estimate the density, which is ultimately what we want to do. Cross-validation cleverly solves this problem spending only one of the points of the sample: One part of the set contains all the points but one, and is used to construct the density, while the other part has a single point that is used to evaluate the density (i.e. the density is computed at this point).

This process is repeated $N$ times, leaving out each point once, and obtaining a corresponding density estimate each time. The contribution of every point to the entropy is then added together to get a final estimate of the entropy. The new expression for the entropy that has to be minimized is:

$$H_S\left(\hat{f}_h\right) = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{(N-1)h^d}.\sum_{\substack{j=1\\ j\neq i}}^{N}K\left(\frac{\vec{x}_i-\vec{x}_j}{h}\right)\right) \qquad (5)$$

Although the phrase "repeated N times" may suggest the opposite, the comparison of equations (4) and (5) shows that the method with cross-validation takes almost the same amount of time.

### 2.1.4 Putting things together: A simple example

Having described a criterion to compare models (cross-validation maximum likelihood, sections 2.1.1 and 2.1.3), and a set of models from which to chose (a family of kernel approximated functions, Section 2.1.2), we reach a point were it would be helpful to present an example showing how density estimation actually works. For the sake of the example, suppose that a density *f(x)* as shown with a black line in Figure 1a has to be estimated from a sample consisting of 200 points, shown in the same figure as vertical red lines. Note that *f(x)* is discontinuous and thus does not belong to the family of functions that can be constructed by placing smooth kernels on finite sample points (is not then in our hypothesis space), but it can be approximated.

The sample is used to define a family of density functions (via the kernel method explained in Section 2.1.2), where each member is characterized by a bandwidth value $h$. Using Equation (5), a score (or entropy value $H$) is computed for each member of the family. Figure 1b shows the correspondence between those values ($h$, the bandwidth, and $H$, the entropy). As explained in Section 2.1.1, the function having the lowest entropy (marked with a red circle in Figure 1b) is found (using classic gradient descent optimization [19]), and selected as the one that best estimates the true density. This function is depicted in red in Figure 1c, along with two other members of the same family of functions: one (in blue) having a smaller bandwidth (blue circle in Figure 1b), which was over-trained/over-fitted to the data, showing excessive oscillations, and the other (in green) having a larger bandwidth (green circle in Figure 1b), was over-smoothed, loosing part of the structure of the true density. The corresponding kernels used for the approximation are shown in Figure 1d.

## 2.2 Quality of the estimates

In the previous section, two qualitatively different but interrelated estimates were obtained: density estimates and entropy estimates. It is our interest in this section to study the quality of both of them. In Section 2.2.1, the discrepancy between the density estimate and the true density is analyzed. Since our goal is to estimate a density, we find particularly important to get a feeling for the kind of errors we should expect. In Section 2.2.2 the error in the score (entropy) is studied. Since this score is used to compare and choose models, the error in the score will clearly affect the chances of selecting a good model.

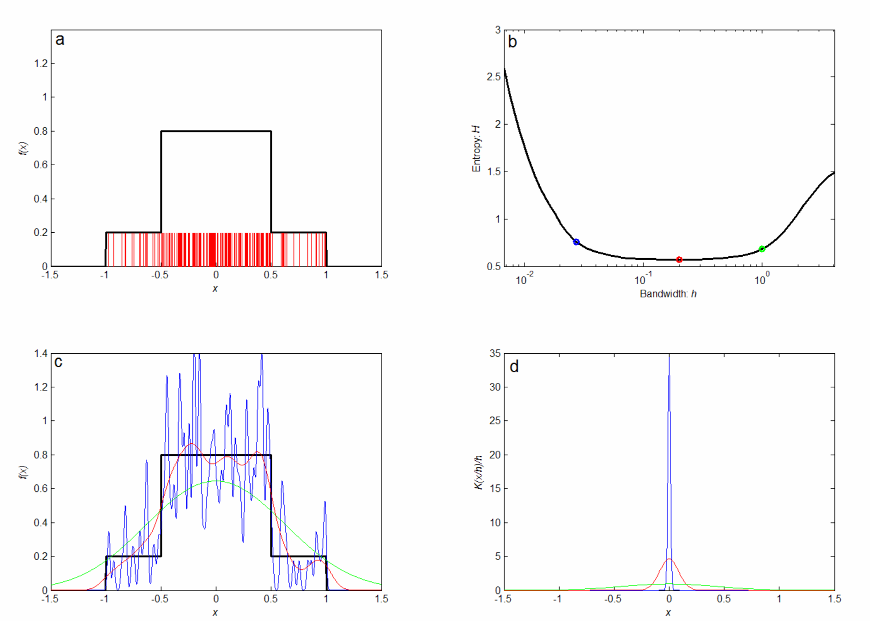

**Figure 1: The density estimation process.** a) The true density and the sample. b) The entropy "landscape". c) Three estimates corresponding to three different bandwidths: the red, green and blue are the best, over-smoothed and under-smoothed estimators respectively. d) The kernels used for each of the estimates in c).

### 2.2.1 The density estimate

An estimate for the density has been obtained in the previous section. But how is it related to the true density? Informally, we could say that the estimate will be a smoothed version of the true density, plus random noise [16]. This can be seen by considering the expectation of the estimate at a single point in the scalar case. By definition, this is not an unbiased estimator of the density. [8] The bias and variance in the density estimate at a point (these are bias is a pointwise measure), could be easily computed [16]:

$$bias_h(\vec{x}) = \frac{h^2\sigma_K^2}{2}\nabla^2 f(\vec{x}) + \text{higher - order terms in } h$$

$$\sigma_K^2 \equiv \int \left\|\vec{t}\right\|^2 K(\vec{t})d\vec{t}$$

$$\text{var}_h(\vec{x}) = \text{var}\left(\hat{f}_h(\vec{x},S)\right) \approx \frac{f(\vec{x})}{Nh^d}\int K(\vec{t})^2 d\vec{t}$$

Let us illustrate these concepts by continuing with the example in Section 2.1.4. To see the variability inherent to the process of drawing ensembles, we repeated 30 times the process carried out in the example above. Each time we drew an ensemble *S*, estimated the best density and plotted the result in Figure 2a. For reference, the true density was plotted in black as before. Notice that each estimate is quite different from the others, but all surround the experimental average (or expected) density, which is plotted in red. As was mentioned before, this density is a smoothed version of the true density. Figure 2b shows a band of width 2 standard deviations around the average density.

[8] In theory, the estimation could be unbiased when $h \rightarrow 0$ and the kernel tends to the delta function or *f(x)* has bounded frequency content and the kernel is an ideal low pass filter in the support of *F(w)*, the Fourier transform of *f(x)*.

The important point to remember from this section is that the (pointwise) bias increases as the bandwidth *h* increases, while the opposite is true for the variance. Cross-validation maximum likelihood is the judge that finds the compromise between the two, keeping the bias as low as possible without increasing too much the variance. To avoid excessive variance then some bias should be introduced in the form of smoothing. This smoothing produces greater deterioration on the fast rising and decaying parts of the density, being those the ones containing higher frequency components (Figure 2b).

### 2.2.2 The entropy estimate

As we saw in Section 2.1.1, the sample *S* has a corresponding entropy value for each model in the hypothesis space. This value can be used, through the maximum likelihood principle, to find the best model in the space. As expected from a score used for comparing models, it is a measure of the *global* fit of each model (recall the discussion about relative entropy at the end of Section 2.1.1). Since we only have limited information about the process (contained in the finite sample available), we do not expect this score to be infallible in discriminating between models. It is only an estimate of the goodness of fit between the model and the process. And as an estimate, it is perturbed by "noise." To illustrate this we return to the example.

To create Figure 2a, multiple ensembles were drawn from the process and corresponding density estimates were computed. Together with each density estimate came a score (the lowest entropy) estimate. These are plotted as blue circles in Figure 3. Using the fact that the distribution of the entropy estimate is asymptotically normal [20], a Gaussian was fitted to them. For comparison, also the true entropy value of the true density was plotted as a black vertical line.

Although all the samples were drawn from the same process, the computed entropies differ from the true entropy. Even more, they do not even surround the true entropy. This bias has its origin in the loss of score (entropy) due to the smoothing applied, and it could be compensated (in theory) using techniques similar to those that we will later develop to asses the variance. But, should it be compensated? We find that it should not, since it corresponds to a real deterioration of the density estimate that should be taken into account when comparing models. Moreover, compensating the score only affects the result of the comparison but does not improve the density estimate.

We also note a curiosity in Figure 3: there are some density estimates that "explain" the data better that the true density. This is because these estimates are only trying to explain the points in the observed ensemble. The red line in the figure corresponds to the entropy of the average

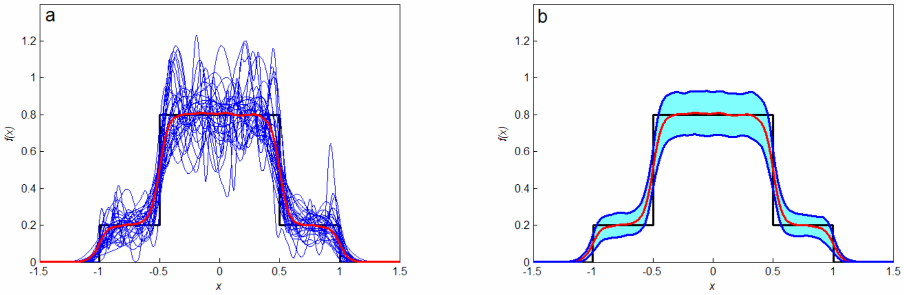

**Figure 2: From each sample a different estimate of the density is obtained.** a) Multiple density estimates (in blue) from multiple samples $S_i$. The average estimate is shown in red. b) A band (in cyan) of two standard deviations around the pointwise average of the estimates (in red).

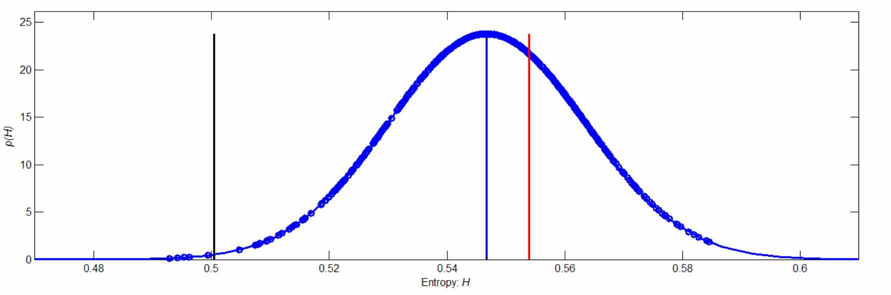

**Figure 3: Each density estimate has a corresponding entropy estimate.** Density of the entropy estimates in blue, true entropy in black and entropy of the average density in red.

density.

How then, can a judgment between models be attributed to an intrinsic difference between the models and not to random effects produced by the noise? We need first an estimate for the variance of the entropy estimate. Joe [21] derived explicit expressions for the bias and variance of this estimate. Unfortunately the true density is required in the computation, making it of little practical use for us, except for providing an idea of the order of convergence of the errors in $N$ (the sample size), $h$ (the bandwidth) and $d$ (the dimension of the domain). A practical alternative is explored in the next section.

### 2.2.3 The bootstrap estimate

*Bootstrap,* [22], is a useful tool to compute error measures for density estimate functionals. The idea is similar to what was done in Figure 3. In order to compute the variability (or even the distribution) of the estimate, we take many samples from the process and compute the function of interest for each one (Figure 4a). The problem with this approach is that usually we only have one sample $S$ from the process to be studied, and it is not possible to get more. The solution proposed by the bootstrap technique is to generate (directly or indirectly) new samples from the original sample $S$, and use these to asses the variability of the functional (figures 4b and 4c). The *classic bootstrap* method works as follows (see Figure 4b):

1. A sample of size $\#S$ is drawn, with replacement, from the original sample $S$.
2. This new sample is used to compute the desired functional (in our case, the sample is first used to estimate a density that is then plugged-in into the entropy functional).
3 Steps 1 and 2 are repeated to obtain many estimates, and in turn an estimate for the variance of the desired functional's estimate (the entropy in our case).

As just presented, this method can not be used in our case, since repeated sample points will force the cross-validation scheme to choose the solution with null bandwidth and deltas in the data points (as explained in Section 2.1.3). Instead, we use a variation of the classical bootstrap technique, known as *smoothed bootstrap* [22, 23]. In this approach, we draw the new sample from the estimate of the density (that we have to find anyway), and not from the original sample itself (see Figure 4c):

1. A density estimate is constructed from the sample, as explained in Section 2.1.
2. A new sample is drawn from this density and used to compute the desired functional; the form of the density estimate makes drawing samples from it particularly easy.
3. As in the classic bootstrap, step 2 is repeated to obtain many estimates that will be used to

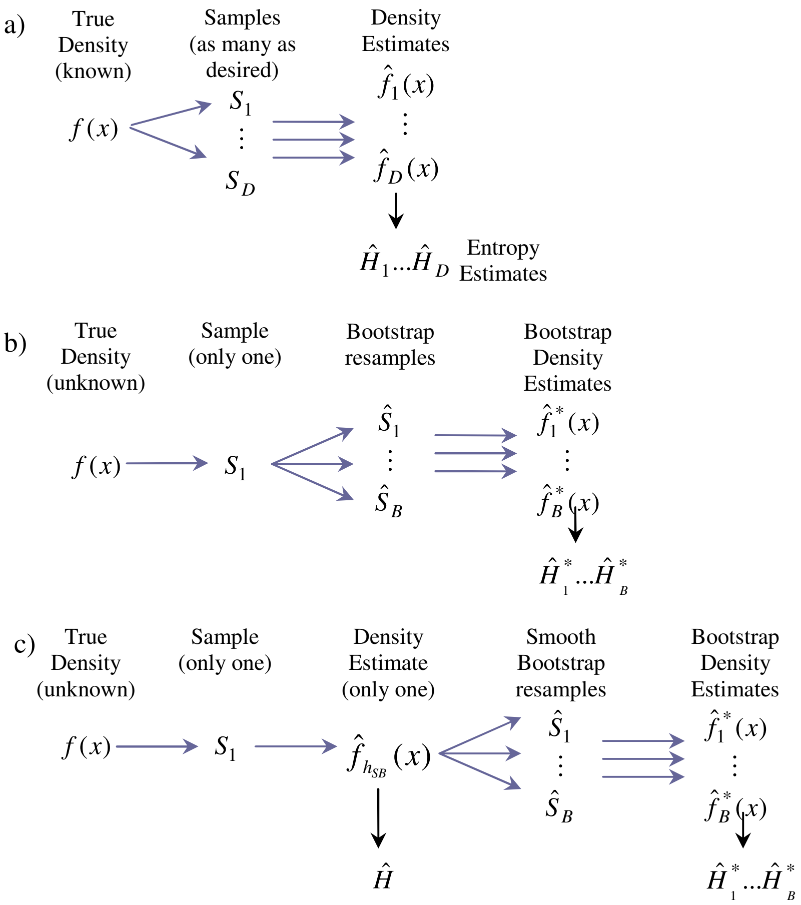

**Figure 4: The Bootstrap and the Smooth Bootstrap.** a) Situation in the example of Figure 3. Since the true density is known, it is possible to draw multiple samples $\{S_1,\dots,S_n\}$, and from each one of them estimate a new density and corresponding entropy. b) Classic bootstrap approach: only one sample $S_1$ is available and the new samples are generated from it. c) Smoothed bootstrap technique: a density estimate $\hat{f}_{h_{SB}}(x)$ is first constructed and then used to draw new samples.

calculate the variance of the desired functional's estimate.

Step 1 is performed only once and the obtained density is used many times in step 2 to draw further samples. The rationale behind the idea of the bootstrap (asses the variability of $\hat{H}$ from that of $\hat{H}^*_1 ... \hat{H}^*_B$) is supported by the fact that $\left(\hat{H}-H\right)$ and $\left(\hat{H}^*_i-\hat{H}\right)$ have the same limit distribution [24] (see Figure 4 for the exact definition of these variables).

Let us use the example started in Section 2.1.4 to exemplify these concepts. Taking advantage of the fact that the true density function is known (in contrast to what usually happens in a real scenario), we can proceed as in Figure 2a to get many density estimates from corresponding ensembles drawn from the process. Then, each estimate (e.g., the *i*-th) is used in two ways (see Figure 5). First, to compute an estimate for the entropy ($\hat{H}_i$) and second to compute a collection of ensembles ($S^1_i ... S^B_i$) that in turn is used to produce new density

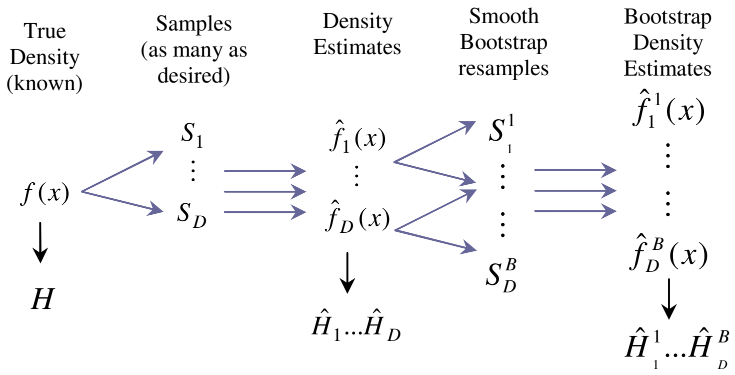

**Figure 5: Simulating the Bootstrap.** Relationship between samples and estimates used in Figure 6. See text for details.

($\hat{f}_i^1(x)...\hat{f}_i^B(x)$) and corresponding entropy ($\hat{H}_i^1...\hat{H}_i^B$) estimates. The idea is to examine the viability of using the variability of $\hat{H}_i^1...\hat{H}_i^B$ to estimate the variability of $\hat{H}_i$. In general we will only have one estimate of the score ($\hat{H}$ in Figure 4c) instead of the many estimates ($\hat{H}_i$ in Figure 5) artificially generated in this case (which was possible because we know the true density). Thus, we will have to resort to bootstrap resamples for an estimate of the variability of the score.

Figure 6a shows the densities for the different scores appearing in Figure 5 for our toy example. The true entropy, which is the entropy computed from the true density, is shown in black. The density of the entropy estimates $\hat{H}_i$, computed from the density estimates $\hat{f}_i(x)$, is shown in blue. The density of the entropy estimates $\hat{H}_i^1$ computed from the bootstrap density estimates ($\hat{f}_i^j(x)$) are shown in red. Figure 6b compares the estimate for the variance of the score $\left(\sigma^2\left(\hat{H}_1^1...\hat{H}_D^B\right)\right)$ to its real value $\left(\sigma^2\left(\hat{H}_1...\hat{H}_D\right)\right)$. This estimate for the variance produced values above the true value but of the expected order.

Setting aside the problem produced by the repeated points in the classic bootstrap resample (which might lead to estimating zero bandwidth kernels), it is not clear whether using the smoothed bootstrap will improve the performance of the estimator over the classic bootstrap in this case, and if it does, how should the smoothing bandwidth ($h_{SB}$ in Figure 4c) be found [25, 26]. It is known that the density that is used to create the resamples in the bootstrap world should be based on a larger bandwidth $h_{SB}$ than the original one (obtained using cross-validation maximum likelihood) [24], but there are no specific rules to choose it. Under these conditions, we decided to use the original bandwidth aware of the fact that this matter requires further research.

In Section 2.3.3 below we will use the smoothed bootstrap to find an estimate of the error in the scores of the models to be compared, and we will take these errors into account to perform the comparison.

## 2.3 Curse of dimensionality

If we had a sample containing a very large number of observations (relative to the dimension

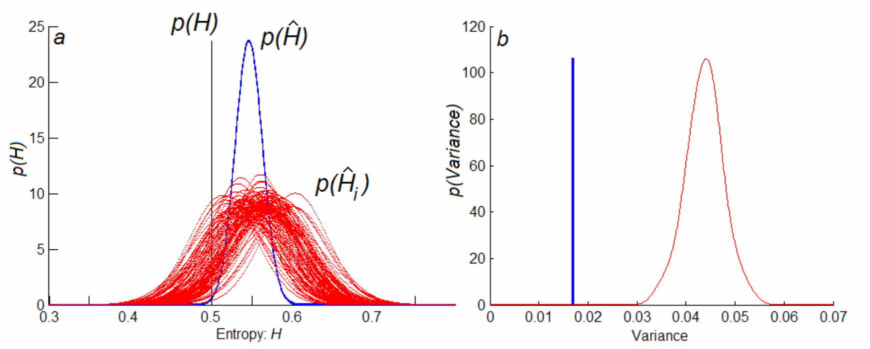

**Figure 6: Densities for the different scores defined in Figure 5 for the toy example.** a) True entropy in black, density of the estimate in blue, and estimates of the density estimate in red. b) True variance in blue and density of the variance estimate in red.

*d* of the space), the problem is solved following the above described framework. Unfortunately, this is not usually the case. Even for very small proteins, the number of degrees of freedom is in the tens or hundreds, and therefore, the number of structures needed to "decently" estimate the density is prohibitively large. This problem is very well known in statistics and has been dubbed the "curse of dimensionality." In this section we suggest a possible solution, which is applicable when the data has a certain property that will allow us to make, for a given error, the sample size virtually independent of the number of residues (degrees of freedom) in the protein. This is one of the main contributions of this article.

So far, any reference to the particular source of the information was avoided, what was described in previous sections is valid for any dataset. However at this point, significant simplifications can be obtained by exploiting an intrinsic characteristic of the ensembles, for example, of folded protein conformations. In the following, we restrict our attention to this kind of dataset, although the same applies to any dataset with the property we introduce in the next paragraph.

An ensemble of folded protein conformations, characterized for example by the backbone torsion angles, has the desirable property that each angle is mainly "related" to a small subset of the other angles, and more important, is "virtually independent" of the rest. What happens to one angle of the conformation is mainly "affected" by the previous and following angles along the chain and perhaps by those angles in its spatial proximity. This is so because a "relation" between angles that are far away will be hard to explain without the inclusion of an angle lying in between (the influence has to be transmitted in some way) and because the number of residues surrounding a given residue is bounded (due to packing considerations). We now formalize these concepts.

### 2.3.1 "Divide and conquer"

We set out to estimate the density $p(x_1, x_2, \ldots, x_d)$[9] of a *d*-dimensional random variable (recall from the beginning of Section 2 that *d*=2(*M*-1), where *M* is the number of residues in the protein). This density can be written as the product of two independent factors,

[9] In previous sections the density to be estimated was called *f*(.) in agreement with the literature on density estimation. However, at this point, where properties of probability densities are being invoked, the use of *p*(.) seems more natural.

$$p(x_1, x_2, ..., x_d) = p\left(x_{i_1} \middle| x_{i_2}, ..., x_{i_d}\right) \,.\, p\left(x_{i_2}, x_{i_3}, ..., x_{i_d}\right)$$

The *i*-subscripts in the right hand side were introduced to account for the fact that any coordinate (arbitrarily chosen) can be "factored out" independently of its position in the original vector. Applying this step inductively, the density can be written as the product of *d* independent factors:

$$p(x_1, x_2, ..., x_d) = p\left(x_{i_1} \middle| x_{i_2}, ..., x_{i_d}\right). \, p\left(x_{i_2} \middle| x_{i_3}, ..., x_{i_d}\right) ... \, p\left(x_{i_{d-1}} \mid x_{i_d}\right). \, p\left(x_{i_d}\right) \qquad (6)$$

By construction and in contrast to Hinton's PoE [10], the factors (experts) in (6) are independent. Since, as stated above, only a few torsion angles (strongly) influence a specific angle, the rest can be discarded from the conditioning set for that particular angle. Assume that we know that any angle is (strongly) influenced by no more than *n* others (with *n* much smaller than *d*), then[10]

$$p(x_1, x_2, ..., x_d) \approx p\left(x_{i_1} \middle| x_{i_1^1}, ..., x_{i_1^n}\right). \, p\left(x_{i_2} \middle| x_{i_2^1}, ..., x_{i_2^n}\right) ... \, p\left(x_{i_{d-1}} \mid x_{i_d}\right). \, p\left(x_{i_d}\right)$$

The original problem of estimating a *d*-dimensional density was reduced to that of estimating *d* independent densities in dimension (*n*+1) or lower. Using properties of the protein conformations, we have then significantly reduced the problem dimensionality (assuming of course that *n* is significantly lower than *d*). For our purposes, it is more practical, and equally valid, to stop the factorization earlier than before and consider the following factorization in (almost) uniform dimension factors:

$$p(x_1, x_2, ..., x_d) \approx p\left(x_{i_1} \middle| x_{i_1^1}, ..., x_{i_1^n}\right). \, p\left(x_{i_2} \middle| x_{i_2^1}, ..., x_{i_2^n}\right) ... \, p\left(x_{i_{d-n+1}}, ..., x_{i_d}\right) \qquad (7)$$

Would it have been better to "factor out" more than one variable at a time? That is, could the following be a better factorization than (7)?

$$p(x_1, x_2, ..., x_d) \approx p\left(x_{i_1}, x_{i_2} \middle| x_{i_{1,2}^1}, ..., x_{i_{1,2}^{n-1}}\right) ... \, p\left(x_{i_{d-n+1}}, ..., x_{i_d}\right) \qquad (8)$$

It is easy to see that no, and thereby (7) is optimal in this sense.

The entropy corresponding to the density in (7) is (see Section 2.1.1 and [15] for the used basic properties of the entropy):

$$H(x_1, x_2, ..., x_d) = H\left(x_{i_1} \middle| x_{i_1^1}, ..., x_{i_1^n}\right) + H\left(x_{i_2} \middle| x_{i_2^1}, ..., x_{i_2^n}\right) + ... + H\left(x_{i_{d-n+1}}, ..., x_{i_d}\right) \qquad (9)$$

From the considerations made in Section 2.1.1, it follows that this is the expression that has to be minimized.

In Section 2.1 we showed how to compute each of the summands in (9).[11] In the next sections we explain how to find the new parameters introduced in this section, namely the *i*-indexes (the conditioned and conditioning variables) and the number *n* of conditioning variables. In this search it will be necessary to access repeated times the values of the entropy summands, for that reason it makes more sense to compute these summands in advance, store them in a database and only then start the search.

It can be fairly objected that since the number of sets of *n* variables grows very fast as *n* increases, this database can be impractically big. Our experiments show that, at least for the tested datasets, the "interesting subsets" of *n* variables can be found incrementally by adding variables to the interesting subsets of (*n*-1) variables. Figure 7a presents an example of this fact, extracted from the dataset used in Section 3.3. This fact has been extensively observed in this

[10] For simplicity we assume the number of conditioning variables (*n*) to be equal in every factor, but this is not strictly necessary.

[11] Actually, we only showed how to compute entropies of the form *$H(x_1,…,x_d)$*. Entropies of the form *$H(x_1|x_2,…,x_d)$* can be computed simply using the relationship: $H(x_1|x_2,…,x_d) = H(x_1,…,x_d) - H(x_2,…,x_d)$.

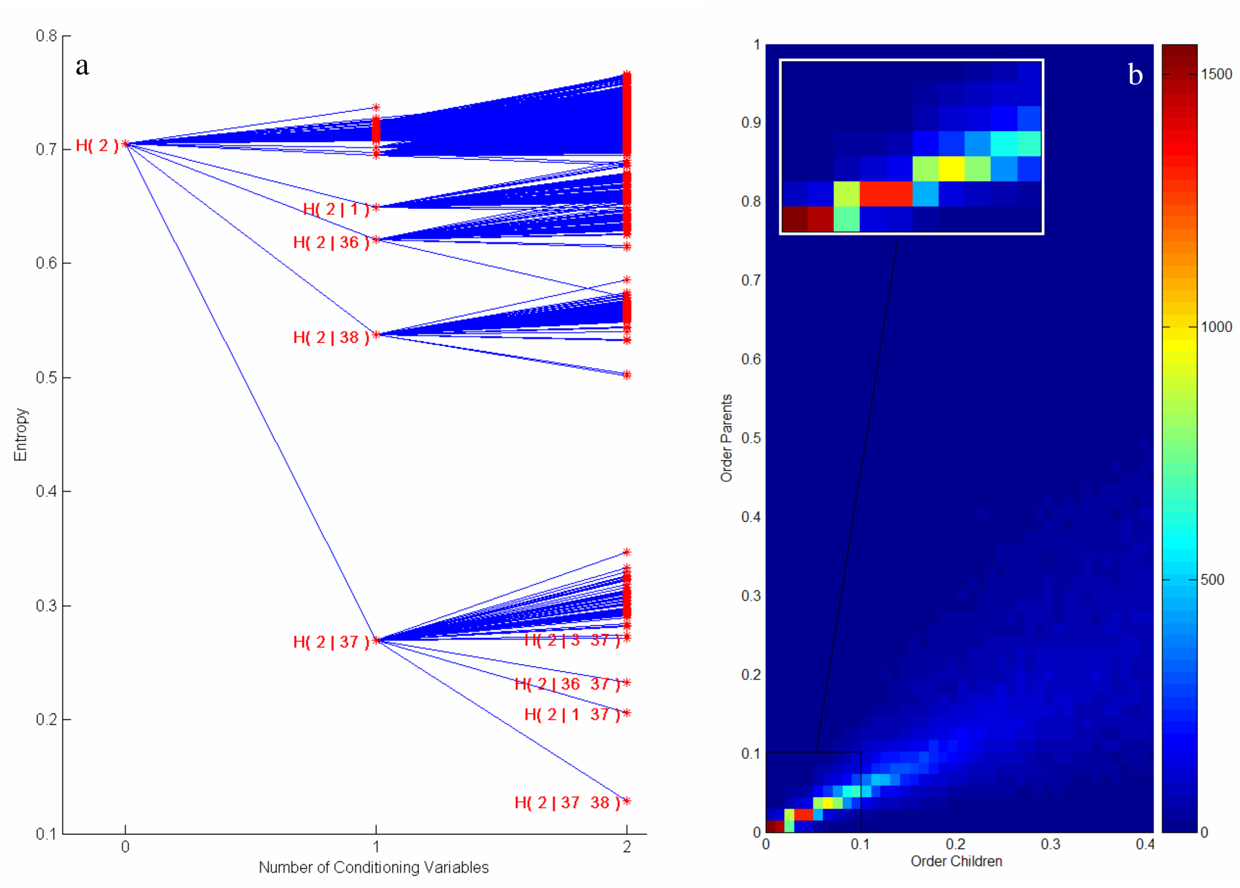

**Figure 7. "Progressive discovery". a)** One example: The best set of two conditioning variables for variable $x_2$ is a superset of the best set of one conditioning variable. In this graph the red stars show the entropy value for the corresponding sets (a selection of them is also named in red). The blue lines connect each conditioning set with its best subset having one less variable. b) 2D-Histogram of points of the form $(I_c, I_p)$ corresponding to the blue lines in Figure 9, where $I_c$ is the index (normalized to be between 0 and 1) of the entropy of the child set in the sorted set of entropies of children and $I_p$ is the index of the entropy of the parent set in the sorted set of entropies of parents. For example, the line connecting H(2| 37 38) and H(2| 37) in a) corresponds to one observation of coordinates (0,0) in this graph, since both sets have the lowest entropy in their respective levels.

dataset and others, and is documented in Figure 7b.

### 2.3.2 Discovering the dependencies

Let us first fix $n$, and find the product of densities of the form in Equation (7) that best explains the data (has the smaller entropy) for that $n$. Let us first introduce some notation to simplify the exposition. Let $I_{1,d}$ be a permutation of the integers from 1 to $d$ (e.g., $I_{1,5} = (3,5,2,4,1)$), $I_{j,d}$ be the indices of that permutation from the position $j$ to $d$ (e.g., $I_{3,5} = (2,4,1)$), and $I_{1,d}^{j}$ the $j$th index of that permutation (e.g., $I_{1,5}^{3} = 2$ for the example above). Also let $C_j^n = \{x_j^1, \ldots, x_j^n\}$ be the set of $n$ coordinates conditioning the coordinate $x_j$ in equations

(7) and (9).

What is left to do then is to find the *i*-indexes subject to the conditions:

1. $(i_1,...,i_d)=I_{1,d}$ : the *i*-indexes must define a permutation of the coordinates (since every coordinate is eventually factored out, but the order is arbitrary); and
2. $C_j^n \subseteq I_{j+1,d}$ : the set of conditioning variables for $i_j$ is a subset of the variables that follow $i_j$ in the permutation (since only those variables not yet factored out can be used to condition). As stated at the beginning of this section, it must contain *n* variables.

Once the permutation $I_{1,d}$ is found (we will explain how we do this below), and assuming that all the terms of the form $H\left(x_{i_j}\middle|C_j^n\right)$ have been previously computed and stored in a database, the conditioning coordinates $C_j^n$ for each conditioned variable $i_j$ can be straightforwardly discovered with the following simple rule: just consider from the set $I_{j+1,d}$ the *n* indexes $\{i_j^1,...,i_j^n\}$ that minimize $H\left(x_{i_j}\middle|C_j^n\right)$. It can be shown that for a given permutation, no other conditioning set can do better. Having established this, the notation can be extended further to rewrite Equation (9) as

$$H(I_{1,d})=H\left(I_{1,d}^1\middle|C_1^n\right)+H\left(I_{1,d}^2\middle|C_2^n\right)+...+H\left(I_{1,d}^{d-n}\middle|C_{d-n}^n\right)+H(I_{d-n+1,d}) \qquad (10)$$

where the $C_i^n$'s are calculated as just explained. This notation stresses the fact that only the permutation has to be found, the other sets follow using the simple rule explained above. Incidentally, we see that $H(I_{1,d})$ is not affected by permutations of the last *n* positions of $I_{1,d}$.

To select a good permutation $I_{1,d}$, a simple "genetic" algorithm based on the ideas of mutation and selection was used. It basically works as follows:

1. Select a random initial permutation $I_{1,d}$ (e.g., $I_{1,5}=(3,5,2,4,1)$, assuming *d*=5 for the sake of the example).
2. Evaluate its entropy, $H=H(I_{1,d})$, using the pre-stored values.
3. Choose two positions at random (e.g., 2 and 4), and swap those coordinates (to obtain $I'_{1,5}=(3,4,2,5,1)$).
4. Evaluate the entropy for the new permutation, $H'=H(I'_{1,d})$.
5. If the new permutation is better than the previous one ($H'<H$), keep it ($I_{1,d}\leftarrow I'_{1,d}$).
6. While the entropy *H* is decreasing and the maximum number of iterations has not been reached, return to step 3.

Two modifications to this algorithm were found to improve its performance. First, only adjacent positions were swapped in Step 3, allowing for a more efficient computation of *H*' in Step 4; and second, the swaps are accepted or rejected with certain probability that depends exponentially in $\Delta H = H'-H$. By its nature, this algorithm is prone to find local minima. To avoid or reduce this problem the algorithm is run many times for each value of *n,* and in each case the best model is kept for that particular *n*.

Intuitively, for small *n*'s, each factor in Equation (7) will be well estimated (since the density being estimated is low dimensional), but the dependencies between variables might be lost. Conversely, for large *n*'s, the dependencies will be captured, but the quality of each factor will be deteriorated. Clearly a compromise must be made. This is detailed next.

### 2.3.3 Selecting the order *n*

In Section 2.2.3 we explained how to use smoothed bootstrap to compute the variance of entropy estimates like the summands $H\left(I^i_{1,d}\middle|C^n_i\right)$ in Equation (10). We first extend this result to compute the variance for the entropy of the permutations, $H\left(I_{1,d}\right)$.

The summands in the right hand side of Equation (10) are independent of each other, and hence the variance of its sum is the sum of the variances of each summand $\sigma^2_{H(I_{1,d})}=\sum_{i=1}^{d-n}\sigma^2_{H\left(I^i_{1,d}\middle|C^n_i\right)}+\sigma^2_{H(I_{d-n+1,d})}$. Also, recall that each summand in Equation (10) is approximately normally distributed, and then its sum also is.

At this point of the algorithm, several models have been found, one for each *n*. Each model has a corresponding score (entropy) as determined by the maximum likelihood principle (see Section 2.1.1), and each score has a corresponding variance, as computed using smooth bootstrap (see Section 2.2.3). To sum up, the results obtained so far look like this:

| *n* | Model | Entropy | Variance |
|---|---|---|---|
| 0 | $p(x_{i_1}).\ p(x_{i_2})\dots p(x_{i_d})$ | $H_0\left({}^0I_{1,d}\right)$ | $\sigma^2_{H_0\left({}^0I_{1,d}\right)}$ |
| 1 | $p\left(x_{i_1}\middle|x_{i_1^1}\right).\ p\left(x_{i_2}\middle|x_{i_2^1}\right)\dots p\left(x_{i_{d-n+1}}\right)$ | $H_1\left({}^1I_{1,d}\right)$ | $\sigma^2_{H_1\left({}^1I_{1,d}\right)}$ |
| ⋮ | ⋮ | ⋮ | ⋮ |
| *m* | $p\left(x_{i_1}\middle|x_{i_1^1},\dots,x_{i_1^m}\right).\ p\left(x_{i_2}\middle|x_{i_2^1},\dots,x_{i_2^m}\right)\dots p\left(x_{i_{d-m+1}},\dots,x_{i_d}\right)$ | $H_m\left({}^mI_{1,d}\right)$ | $\sigma^2_{H_m\left({}^mI_{1,d}\right)}$ |

Which one of the *m* models is best? According to the maximum likelihood principle, the one that has the lowest entropy. But we assigned to the models, via the smoothed bootstrap, not just a score, but a probability density of scores. We have then to define a criterion to choose between those models.

Assume we have two models A and B with $n_A$ and $n_B$ conditioning variables respectively, $n_A > n_B$. Assume also that the density of the entropy estimate for each model is as shown in Figure 8. Which model is better? Intuitively we should choose model A, since the probability that it performs better than model B (again according to maximum likelihood), is higher than the probability of the reverse case. Obviously if the opposite were true, we should choose model B. If neither one of the possibilities is true, it makes sense to choose the computationally less expensive option, meaning the model with the smallest number of conditioning variables. Formalizing this, we end up with a selection rule (for $n_A > n_B$):

$$P(H_A < H_B) > kP(H_A \geq H_B) \Leftrightarrow P(H_A < H_B) > \frac{k}{1+k}$$

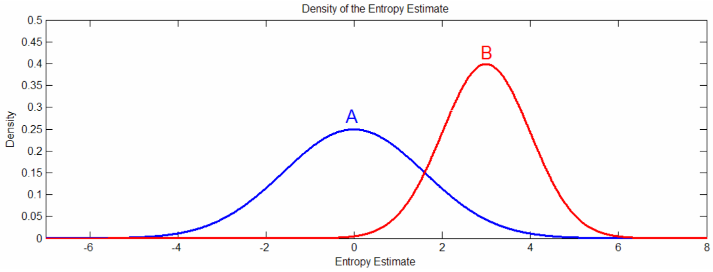

**Figure 8: Choosing the model's order (*n*).** Density of the entropy estimates for two hypothetical models A and B. Which one should be chosen?

where $k$ is a parameter introduced to account for the fact that model A is computationally more expensive and should be conveniently adjusted.

Knowing that the densities of the entropy estimates are (approximately) Gaussian with mean and variance given by the two last columns of the previous table respectively, the probability can be computed (for instance using Monte Carlo), and the condition evaluated for the models, resulting in a unique model being selected as the best representative for the data. This model may not be sufficient, but it is the best that can be obtained with the available data according to our optimality criterion. This is an important concept: our proposed framework finds the best possible model (with respect to the selected optimality criterion), and if this one is not good enough, more data will have to be collected, but the data was used efficiently.

Figure 9 summarizes the procedure explained in this section. This concludes the derivation of the computational framework. We proceed now to its validation and application to real data.

# 3 Results and applications

In order to test our method, we start with a set of artificially constructed examples. In Section 3.1 we present these results. In Section 3.2 we apply the method to two real datasets. First an ensemble of conformations of the 12 residues long β-hairpin tryptophan zipper [27] is analyzed (Section 3.2.1). Then in Section 3.2.2, the villin headpiece [28], a significantly more complex peptide having 36 residues is studied.

## 3.1 Validation via artificial examples

Four artificial datasets with different dependency levels were constructed to validate the proposed method. Those are schematically represented on the left of Figure 10. In all these datasets the sample size ($N$) was 500, and the dimension of the data points ($d$) was 6. The method explained in Section 2 was applied to each dataset. In all cases the dependencies were correctly discovered. The right side of Figure 10 shows the evolution of the entropy as more conditioning variables are allowed in the model for each dataset. In each case, the correct number of conditioning variables ($n$) was found. It is worth noting that contrarily to what is expected for an infinite dataset, adding unnecessary conditioning variables deteriorates the model. For finite sample sizes, the price to pay for considering more dependencies is more smoothing, which in turn deteriorates the model. As explained before, a compromise must be made, and this is done here automatically, by selecting the optimum $n$ using the rule described in Section 2.3.3 and the

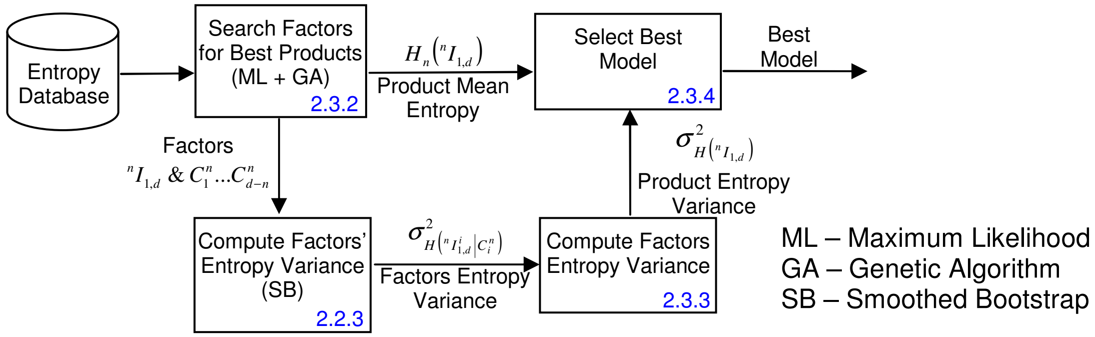
Entropy Database
Search Factors for Best Products (ML + GA)
2.3.2
$H_n\left({}^nI_{1,d}\right)$
Product Mean Entropy
Select Best Model
2.3.4
Best Model
Factors
${}^nI_{1,d}$ & $C_1^n \dots C_{d-n}^n$
$\sigma^2_{H\left({}^nI^i_{1,d}|C^n_i\right)}$
Factors Entropy Variance
$\sigma^2_{H\left({}^nI_{1,d}\right)}$
Product Entropy Variance
Compute Factors' Entropy Variance (SB)
2.2.3
Compute Factors Entropy Variance
2.3.3
ML – Maximum Likelihood
GA – Genetic Algorithm
SB – Smoothed Bootstrap

**Figure 9: The complete procedure explained in Section 2.3.** The blue numbers in the corner of each box correspond to the sections in which the particular computational box is explained.

information in the graphs of Figure 10.

## 3.2 Protein examples

### 3.2.1 β-hairpin tryptophan zipper

Our first real ensemble consists of 481 conformations of the β-hairpin tryptophan zipper [3, 6, 27], a peptide having 12 residues. The backbone of these peptides can be described by 22 torsion angles (11 $\phi$'s and 11 $\psi$'s), and consequently we need to estimate a 22 dimensional probability distribution function (pdf).

In Figure 12a, the evolution (with respect to $n$, the number of conditioning variables) of the entropy is displayed. Note that including more than two conditioning variables in the density factors does not significantly improve the estimate (compared to three), and can even deteriorate it (compared to four). As explained in Section 2.2.2, the magnitude of the improvement/deterioration should be judged relative to the variance corresponding to the score of each model (represented by the blue band in Figure 12a).

Applying the algorithm of Section 2.3.3, we select the model with two conditioning variables as the one that best represents the ensemble. We do not claim that the true dimensionality of the process is two, but only that for the current available sample, the benefit obtained in computing a pdf using the additional dependencies accounted for when including

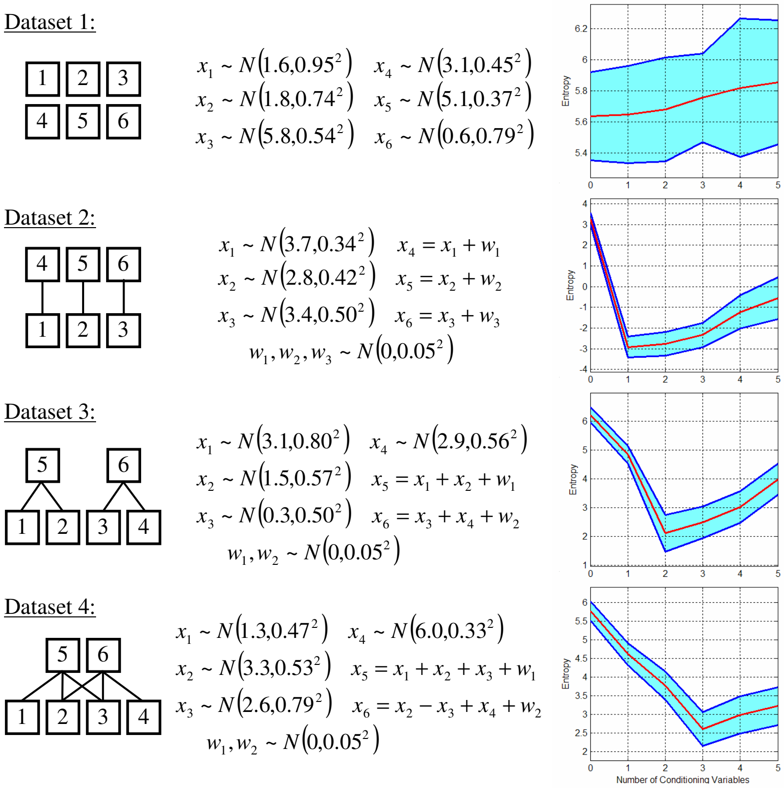

**Figure 10: Validation with four artificially generated datasets.** Left: the relationship between the variables is schematically represented. Middle: a formal description of the relationships. Right: evolution of the entropy as more conditioning variables are included. The cyan band represents the 99% confidence interval.

more than two conditioning variables, is smaller that the harm done by the additional smoothing required.

It is interesting to observe the dependencies between variables found by the algorithm, Figure 11. Notice that as expected, the angles are often conditioned on the adjacent (along the chain) angles of the same kind ($\phi$ or $\psi$), or on the corresponding angle of the opposite kind. This is further evidence that the algorithm is doing what it should. An interesting fact to note is that we find more frequent conditioning on the $\psi$'s; this can be explained by the asymmetric roles of $\phi$ and $\psi$ in the Ramachandran map. It is appealing that this effect emerges from the combination of the formalism and the simulation data, rather than something which must be included by hand.

We can gain insight of the structure of the ensemble by explicitly computing the density value in the available observations using the model selected by our framework. Figure 12b presents a plot of these values sorted from the least likely to the most. It can be seen in this figures that a few conformations are much more likely than the rest (note that the log-density is plotted) and a few conformations are much more unlikely than the rest. The experimentally determined structures (PDB [29] entry 1LE0), also shown in the figure, can be seen to have similar probability densities located between these two extremes.

Are those conformations similar to each other, or more precisely, are there many modes in the density or just one? To consider this question, which needs explicit pdf estimation as done here, we plotted in Figure 12c the distances between all the conformations. [12] Since the conformations are sorted (as explained above), the distances between the most probable conformations lie in the upper right corner. The pattern in this area agrees with a unimodal

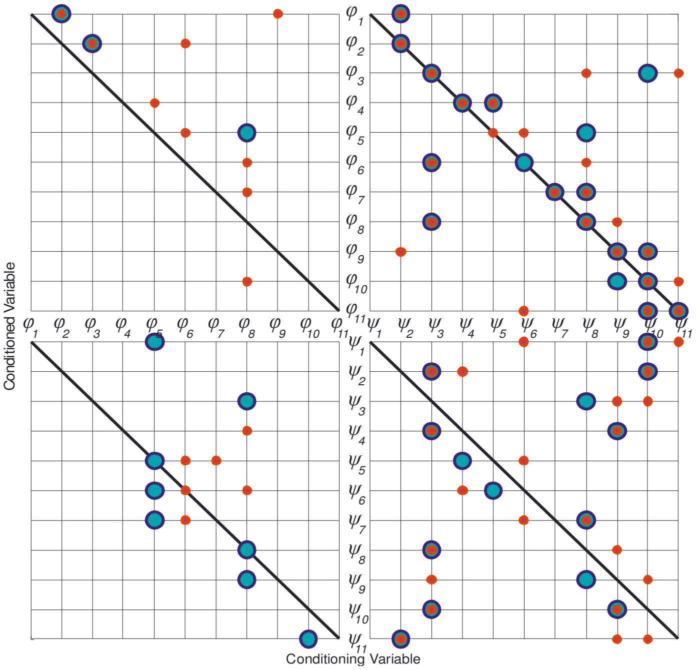

**Figure 11: Dependencies discovered between the variables.** Dependency diagram when two (in blue) and three (in red) conditioning variables are allowed. The conditioning variables for a given variable appear as dots in the corresponding row. For instance, when two conditioning variables are allowed for $\phi_2$ (2nd row in the topmost squares), $\phi_3$ (3rd column in the leftmost squares) and $\psi_2$ (2nd column in the rightmost squares) are selected. If one more variable is allowed, $\phi_6$ is also included.

[12] T are two

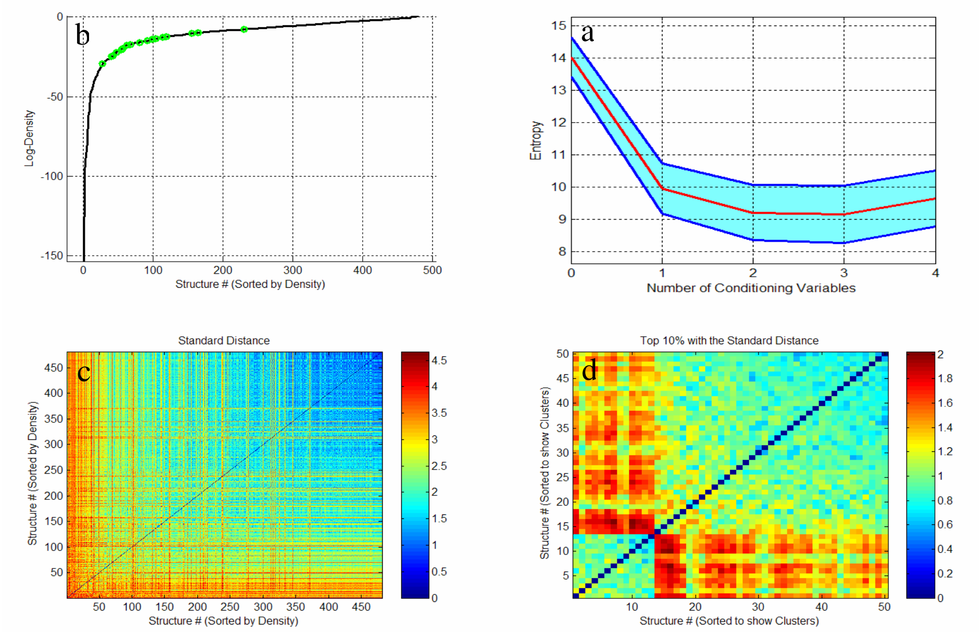

**Figure 12: Analysis of the estimated density.** a) Evolution of the entropy as more conditioning variables are included. As before, the band represents the 99% confidence interval. b) Logarithm of the density for each structure (observation). The structures are sorted from the least probable (on the left) to the most probable (on the right). The experimentally determined structures (PDB entry 1LE0), which were not part of the set used for deriving the pdf, are shown in green. c) Distances between all the structures sorted as in b. d) Distances between the 10% most probable structures sorted to show clusters. Use the color bar provided to translate into the corresponding numeric values.

density. This suggests a way to choose a representative for the ensemble, if one needs to be selected, simply as the most probable conformation. Zooming in on the top ten percent of the structures however, shows that these structures cluster around two distinct modes (Figure 12d), that lie relatively close to each other.

The three most likely conformations and the three least likely conformations from the available ensemble are shown in Figure 13. As expected, the most likely conformations have more hydrogen bonds and hence are more stable.

### 3.2.2 Villin headpiece

The second real ensemble we analyzed consists of 1543 conformations of the villin headpiece molecule [3, 6], a peptide having 36 residues (70 torsion angles). The same tests described in the previous section were performed on this dataset and similar conclusions can be drawn. Figure 14 shows the corresponding results. In this case, since the sample is more than three times bigger than in the previous example (1543 versus 481), the system is able to capture three conditioning variables instead of two (see Figure 14b). As before, few conformations are

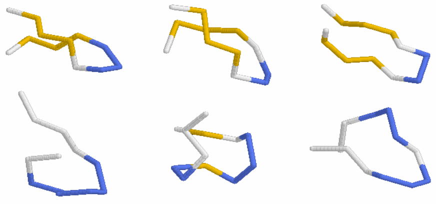

**Figure 13: The most likely and unlikely conformations.** The three most likely (top) and less likely (bottom) conformations according to the density computed by our proposed framework.

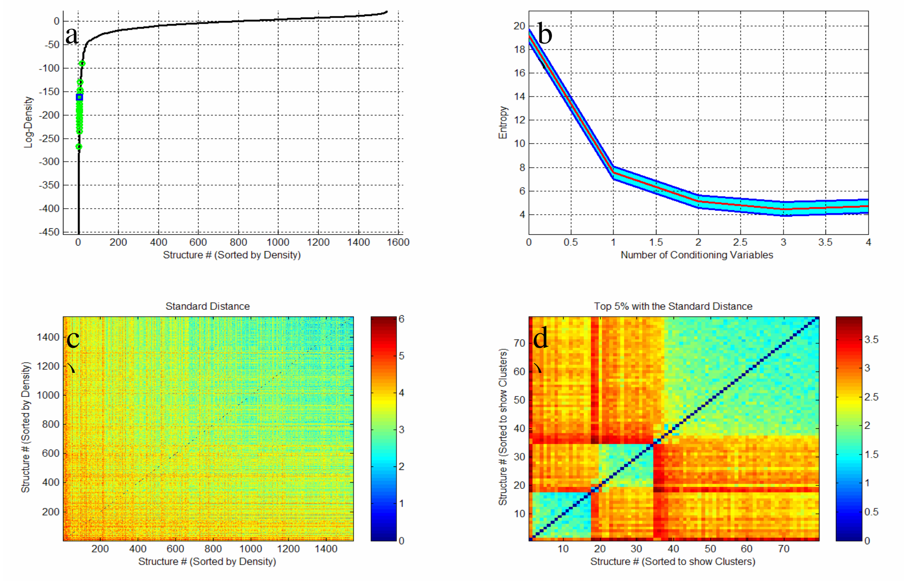

**Figure 14: Analysis of the estimated density for the second dataset.** a) Logarithm of the density for each simulated (black line) and the experimental (green circles) structure. The structures are sorted from the least probable (on the left) to the most probable (on the right). The density of the average "native" is indicated by a blue square. b) Dependency of the entropy on the number of conditioning variables. c) Distances between all the structures sorted as in a. d) Distances between the 10% most probable structures sorted to show clusters. Use the color bar provided to translate into the numeric values.

much more likely than the rest (Figure 14a) and those are situated in what appears to be the mode of a unimodal density (Figure 14c). However, when this mode is carefully examined, it splits into several modes (Figure 14d).

For this molecule, an ensemble of structures determined using NMR techniques can be found in [30], together with its minimized average ("native") (PDB entry 1VII). We can estimate the probability density of these structures and compare it with the probability density obtained

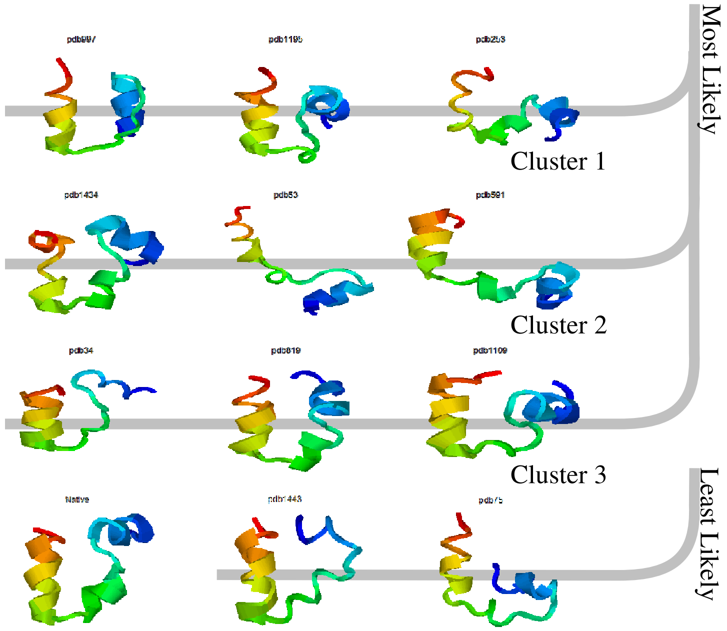

**Figure 15: Selected conformations from the ensemble.** The most probable conformations of the three (most probable) clusters of figure 14d are in the first three rows, one cluster per row. The two least probable conformations of the whole ensemble and the "native" appear in the fourth row.

for the other structures in the ensemble. By doing this we found that the experimental ensemble (circles and square in Figure 14a) belongs to the group of most unlikely structures. This might indicate that the ensemble of simulated structures is not yet correctly capturing the whole information about the native state[13] and also raises the question of how much confidence should be given to a single (experimental) structure. The most and least probable conformations are shown in Figure 15.

To further study this phenomenon of low probability associated with the native experimental structures (extracted from the work of McKnight et al. [30]), we plotted the value of each computed factor (from Equation (7)) in its corresponding spatial location (Figure 16). In the same figure we included a graph of the factors where the "native" structure appears to differ the most from the simulated structures, resulting in low factor values, and thereby small probabilities. For the sake of visualization, and since according to Figure 14b not much explicative power is obtained when using more than one conditioning variable, we only included one conditioning variable in the graphs of the factors.

The technique presented in this work can not only be used to asses the "probability" of an existing structure, but also to generate novel structures having (presumably) high probability. To obtain these structures we start by choosing a point in the space of conformations and follow the direction of the gradient of the computed pdf, until a local maximum is reached. In Figure 17, left, the change of the probability is shown for three different groups: those that started at the most unlikely conformations of the original ensemble (in red), those that started at the most likely conformations of the original ensemble (in blue) and those that started at conformations of the original ensemble having intermediate probability (in green). It can be seen in the figure that, in every case, but especially for the red (most unlikely) structures, there was a marked "improvement" in the probability of the structures. Why is the probability increasing? One explanation is that the optimization is assembling together "popular" parts to create the new structures, finding the consensus among the observed structures for each region of the protein. These new conformations automatically obtained from the computed pdf can be used for example as novel initial conditions for molecular dynamics or for producing new candidates for high resolution protein design.

It is also of interest to asses the "nativeness" of the generated structures. As we repeatedly mentioned in this work, see Introduction, using a single structure to characterize the native state may not be the best approach. In this case, the distance from the new structures to the experimental ensemble (Figure 17, center) is of the same order as the internal variability of the experimental native observations themselves (Figure 18). Lacking the necessary observations of the experimental structure to follow the approach introduced in this article, we have to resort instead to the distance (11) below to the "native" ensemble as a measurement of nativeness (the distance to the ensemble is given by the minimal distance to all the elements in it). These results are plotted in Figure 17, center. Note that the distance for the majority of the most likely (blue and green) structures is "improved" by the "optimization." In Figure 17 right, where we plotted the log-probability of the new structures versus their distance to the native ensemble, it can be seen that there is a tendency for the closest structures to the native ensemble to be the most likely. Thereby, the pdf is finding from all the molecular dynamics results, the "best" ones according to this distance. These "best" ones then lead to new conformations, new samples of the conformation space, via the pdf gradient ascent technique mentioned above, Figure 17 left.

[13] Possibly due to an inaccurate force field used in the simulation.

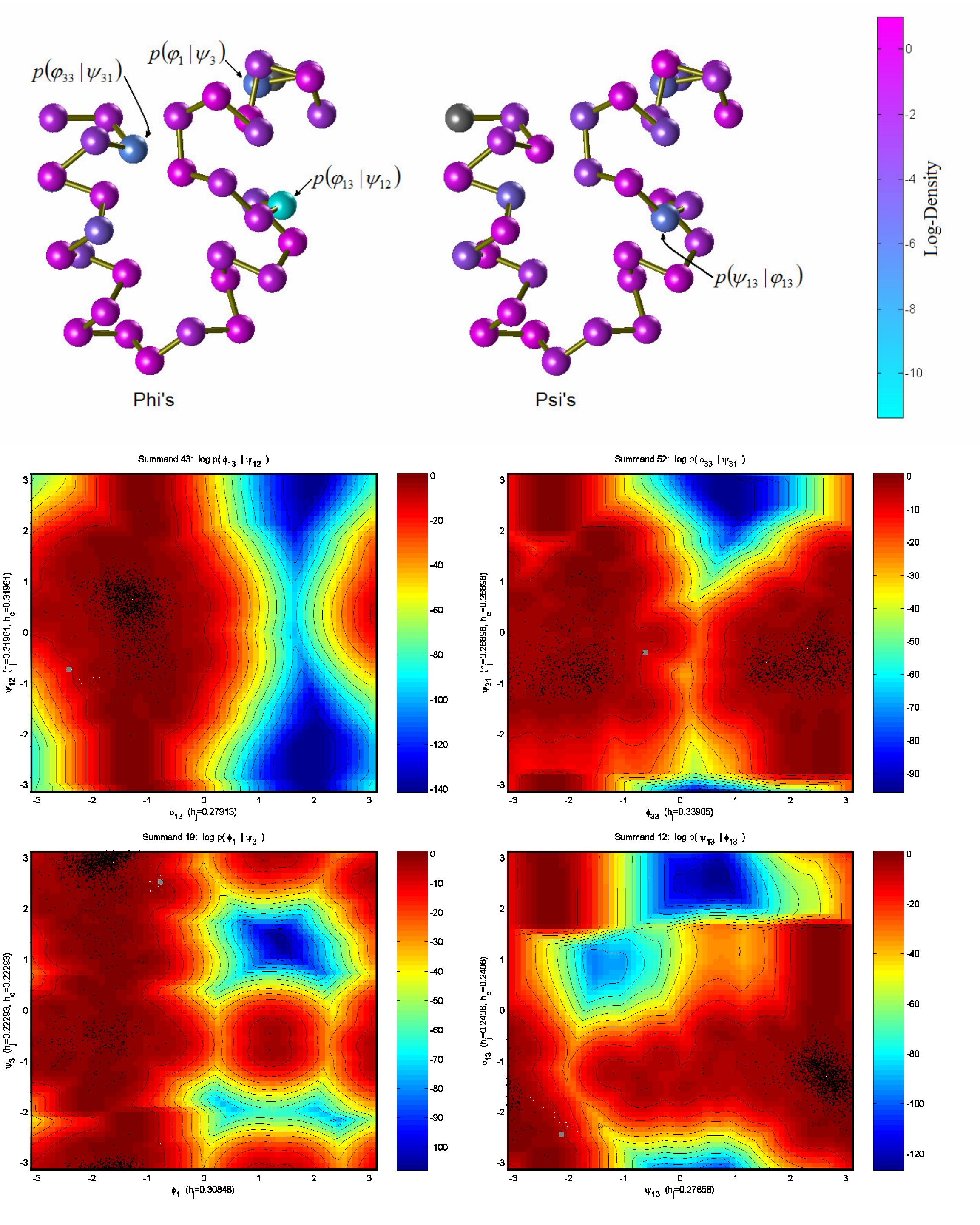

**Figure 16: Contribution of each factor to the overall density of the average native.** Top: The density of each factor ($\phi$'s on the right, $\psi$'s on the left) is represented by the color of the corresponding ball. The four lowest factors are labeled and a graph is included at the bottom. Bottom: Graph of the log-density for selected factors. Black dots represent the molecular dynamics' samples used to compute the density; white circles represent the experimental structures; gray square is the "native" (experimental average) structure; white 'x' and '*' represent the least and most likely structures of the molecular dynamics' ensemble respectively. Note that the native has some factors not located at the top of the density, thereby explaining why the overall probability of this single structure is low.

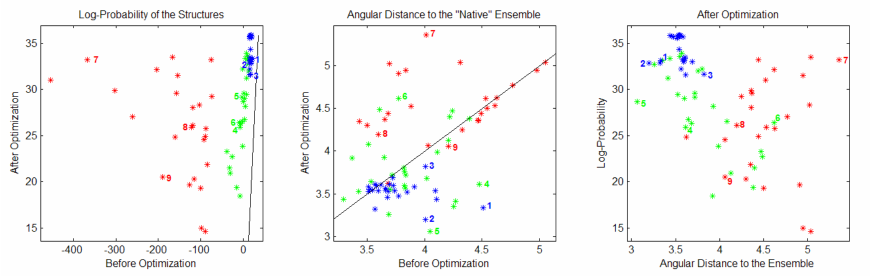

**Figure 17: New structures**. Those were obtained by gradient ascent starting at the most unlikely conformations of the original ensemble (in red), the most likely conformations of the original ensemble (in blue), and original conformations of intermediate likelihood.

# 4 Conclusions and discussion

A method to estimate a probability density function in the space of folded protein conformations was developed. This method does not have any free parameters to fix other than the shape of the kernel used (Section 2.1.2), and relies on fundamental results from estimation and information theory and on the assumption that only a few angles strongly influence a specific torsion angle (or in general, a few variables strongly affect another specific variable). This is needed in order to reduce the dimensionality of the problem. With our framework, we not only obtain the best possible pdf (modulo our optimality criteria), but also learn the order of the model (*n*) and explicitly find the torsion angles (variables) dependencies.

Better estimates might be obtained if constraints between the angles (e.g., the allowed regions of the Ramachandran plot) and/or energy priors (i.e. Boltzmann weighting) are included in the model. Obviously this can be straightforwardly done and improvements are expected. Other acknowledged places for improvement include the optimization procedure used to choose the bandwidth for the kernels (which may be substituted by more efficient methods); the Fast Gauss Transform or Dual Tree methods can be used to speed up the computation of the entropies [31, 32]; and the genetic algorithm explained in Section 2.3.2 to discover the dependencies. Also, more research is required for selecting the bandwidth used in the smooth bootstrap step (Section 2.2.3), unless this problem is altogether avoided by using other methods (e.g., the jackknife [22]) which do not produce resamples having repeated observations.

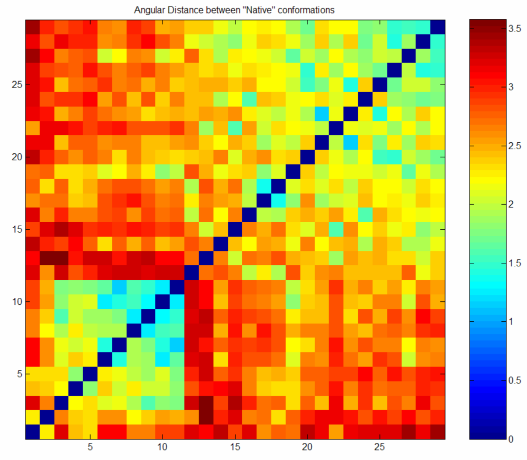

**Figure 18: Distance matrix between experimental conformations.** Angular distance between the experimental structures (from [32]) computed using (11).

Having a method to statistically characterize the folded ensemble, it could be tempting to apply the same method to other ensembles as well. Recall that in order to apply this proposed method, the ensemble must satisfy the basic assumption about the dependency between the torsion angles (or the variables used for the structure representation in general). In particular, for the ensemble of partially folded conformations, this assumption is less valid due to the diversity of long range interactions that can take place, being the number of dependencies *n* that are to be taken into account in this case quite large, and thereby, not obtaining a dimensionality reduction as significant as the one obtained in the folded case. Again, this method can be easily extended to other descriptions of proteins such as those including the side chains or even to completely different datasets, if those satisfy the basic assumption stated above. Furthermore, we expect better results if more complete descriptions are supplied (e.g. the side chain angles/pair-wise distances are included), since those can also be included as conditioning variables, if they turn out to be the most informative.

Our results suggests that for a given accuracy, the number of sample points in the ensemble does not need to grow exponentially with the number of residues in the peptide, but only with the number of those actually affecting each other. In other words, the "true" dimension of the dataset is much smaller than the total number of torsion angles, being defined only by the interactions in small neighborhoods and not across the whole protein.

When comparing structures in order to validate our technique (Section 3.2.2), we used the angles distance (which, by the way, is intimately related to the Von Mises kernels):

$$d(\vec{x},\vec{y}) = \sqrt{\sum_{i=1}^{d}\{1-\cos(x_i - y_i)\}} \qquad (11)$$

The choice of this metric is somehow arbitrary, undermining the results derived using it. This choice of metric is not completely compatible with the density estimate proposed in this work. If we were to believe that Equation (11) is the right metric to use for this space, we should have chosen symmetrical kernels in *d*-dimensions. Conversely, since as explained before, this type of kernels is a very bad choice, we think that using the metric of (11) is not optimal. In Section 2.1.2, while presenting the kernel density estimation technique, a distance was implicitly used to asses the contribution that each observation has on the point where the density is being estimated. Now that a density has been found, it may be interesting to use this relationship between distances and densities in the other direction, to derive a "distance" *from* the density. This natural "distance" takes into account the structure of the ensemble, and this will be further studied elsewhere.

Since it is commonplace in this field to use the $C_\alpha$-RMSD to compare conformations, for completeness, we conclude by including in Figure 19 the equivalent of figures 17b-c and 18 computed using this metric instead of (11). This approach presents two main difficulties. First, it is not trivial to compute the 3d coordinates of the $C_\alpha$'s from the torsion angles (we use the standard Engh and Huber [33] angles for the reconstruction). Second, small variations in the torsion angles can produce large variations in the 3d coordinates. As before, we compute the RMSD distance to the whole experimental native ensemble, not just to an average structure. Note once again the automatic clustering of the most probable conformations, as computed with our pdf, and how it gets closer to the experimental ensemble. Note also the large inner variability (left figure) among the experimental conformations themselves, of the same order of the variability of the new conformations created by our algorithm when starting from the most probable ones. Of course, if this kind of comparison was intended, a different set of features should have been chosen in the first place (not the torsion angles).

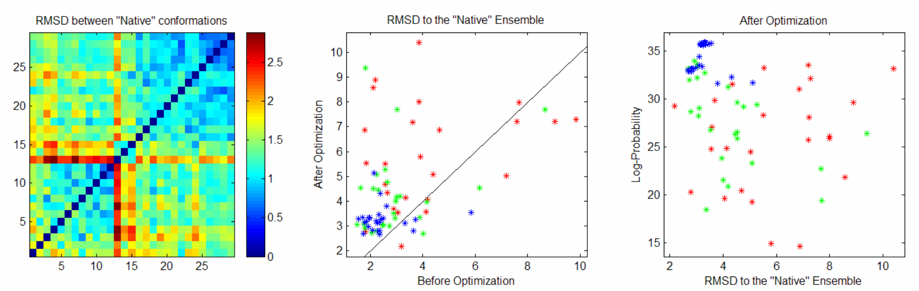

**Figure 19:** Graphs corresponding to figures 17b-c and 18 but computed using $C_\alpha$-RMSD.

To conclude, based on statistics and information theory, we have presented a framework to compute the distribution of protein conformations, with possible applications from protein comparisons to conformation space sampling to high resolution protein design. These applications, as well as the exploitation of the pdf to define new distance functions (to be reported elsewhere), may promote a shift from the current emphasis on single structures to the consideration of whole ensembles, allowing all the available information to play a role.

**Acknowledgements:** We thank Bojan Zagrovic for the carefully reading of the paper, his suggestions (especially those resulting in the inclusion of figures 17 and 19) and for pointing out relevant references. We also thank David Baker, Alexander Grossberg and Birgit Grund for helpful discussions, and ONR, NSF, NGA and DARPA for the founding. This work was carried out in part using computing resources at the University of Minnesota Supercomputing Institute.

# 5 References

1. Rieping W, Habeck M, Nilges M. (2005) Inferential structure determination. Science 309: 303-306.

2. Rother D, Sapiro G, Pande V. (2005) Statistical characterization of protein ensembles. RECOMB 2005 Poster Abstracts: 297-298.

3. Zagrovic B, Snow CD, Khalid S, Shirts MR, Pande VS. (2002) Native-like mean structure in the unfolded ensemble of small proteins. Journal of Molecular Biology. 323: 153-164.

4. Shortle D, Simons KT, Baker D. (1998) Clustering of low-energy conformations near the native structures of small proteins. Proceedings of the National Academy of Sciences, USA. 95: 11158-11162.

5. Bradley P, Misura, K. M. S., Baker D. (2005) Toward high-resolution de novo structure prediction for small proteins. Science 309: 1868-1871.

6. Pande VS, Stanford University. (2005) Folding@home distributed computing. Available: http://folding.stanford.edu/ via the Internet.

7. Baker D. (2003) The baker laboratory. Available: http://www.bakerlab.org/ via the Internet.

8. Teague SJ. (2003) Implications of protein flexibility for drug discovery. Nature Rev Drug Discov 2: 527-541.

9. Branden C, Tooze J. (1998) Introduction to protein structure. New York: Garland Publishing, Inc.

10. Hinton GE. (2002) Training products of experts by minimizing contrastive divergence. Neural Computation 14(8): 1771-1800.

11. Akaike H. (1974) A new look at the statistical model identification. IEEE Transactions on Automatic Control AC-19: 716-723.

12. Burnham KP, Anderson DR. (1998) Model selection and inference: A practical information-theoretic approach. New York: Springer-Verlag. : p. 353.

13. Viola PA. (1995) Alignment by maximization of mutual information. PhD Thesis, MIT.

14. Jaynes ET. (1957) Information theory and statistical mechanics. Phys Rev 106(4): 620-630.

15. Cover TM, Thomas JA. (1991) Elements of information theory. John Wiley & Sons, Inc.

16. Silverman BW. (1986) Density estimation for statistics and data analysis. New York: Chapman and Hall.

17. Mardia KV, Jupp PE. (2000) Directional statistics. John Wiley & Sons, Ltd.

18. Bowman AW, Azzalini A. (1997) Applied smoothing techniques for data analysis. New York: Oxford University Press.

19. Boyd S, Vandenberghe L. (2004) Convex optimization. Cambridge, UK: Cambridge University Press.

20. Cuevas A, Romo J. (1997) Differentiable functionals and smoothed bootstrap. Ann Inst Statist Math 49: 355-370.

21. Joe H. (1989) Estimation of entropy and other functionals of a multivariate density. Ann Inst Statist Math 41(4): 683-697.

22. Efron B, Tibshirani RJ. (1993) An introduction to the bootstrap. Chapman and Hall.

23. Cheng RCH. (1995) Bootstrap methods in computer simulation experiments. Proceedings of the 1995 Winter Simulation Conference : 171-177.

24. Alonso AM, Cuevas A. (2003) On smoothed bootstrap for density functionals. Nonparametric Statistics 15: 467-477.

25. Hall P, DiCiccio TJ, Romano JP. (1989) On smoothing and the bootstrap. The Annals of Statistics 17(2): 692-704.

26. Silverman BW, Young GA. (1987) The bootstrap: To smooth or not to smooth. Biometrika 74: 469-479.

27. Snow CD, Qiu L, Du D, Gai F, Hagen SJ, et al. (2004) Trp zipper folding kinetics by molecular dynamics and temperature-jump spectroscopy. PNAS 101(12): 4077-4082.

28. Zagrovic B, Snow CD, Shirts MR, Pande VS. (2002) Simulation of folding of a small alpha-helical protein in atomistic detail using worldwide-distributed computing. J Mol Biol 323: 927-937.

29. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, et al. (2000) The protein data bank. Nucleic Acids Research 28: 235-242.

30. McKnight J. (2001) McKnight lab PDB files. 2005: 1. Available: http://people.bu.edu/cjmck/pdb.htm via the Internet.

31. Elgammal A, Duraiswami R, Davis LS. (2003) Efficient kernel density estimation using the fast gauss transform with applications to color modeling and tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(11): 1499-1504.

32. Gray AG, Moore AW. (2003) Nonparametric density estimation: Toward computational tractability. International Conference on Data Mining .

33. Engh RA, Huber R. (1991) Accurate bond and angle parameters for X-ray protein structure refinement. Acta Cryst A47: 392-400.

34. Beirlant J, Dudewicz EJ, Györfi L, Van der Meulen, E.C. (1997) Nonparametric entropy estimation: An overview. International Journal of Mathematical and Statistical Sciences 6(1): 17-40.